**Searle on Programs and Intentionality** Sharvy, Richard Canadian Journal of Philosophy ; Supplementary Volume; Jan 1, 1985; 11, ProQuest pg. 39

*CANADIAN JOURNAL OF PHILOSOPHY*
*Supplementary Volume 11*

# *Searle on Programs and Intentionality**

RICHARD SHARVY
2447 Kincaif Street
Eugene, OR 97405
U.S.A.

> ... it is possible for a man to write a piece correctly by chance or at the prompting of another: but he will be literate only if he produces a piece of writing in a literate way, and that means doing it in accordance with the skill of literate composition which he has in himself.
>
> Aristotle, *Nicomachean Ethics* II 4 1105a23-35

---

* I thank Daniel Dennett, Edward Erwin, Dale Jamieson, Cristelle Leaf, William G. Lycan, Alice Perrin, and Morton Winston for their good conversation and correspondence on many of these problems. A version of this paper was read to the Florida Philosophical Association in November 1981.



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

> A man blind from birth might produce syllogisms about colours, but he is reasoning only about names to which nothing in his mind corresponds.
> *Physics* II 1 193a7-9

John Searle has published an exciting discussion of the relations between computer programs and intentionality. His Chinese Room Example shows a principle of non-sufficiency, namely that

NAS: instantiating a computer program is not always by itself a sufficient condition of intentionality.

From this, Searle infers that 'The *explanation* of how the brain produces intentionality cannot be that it does it *by* instantiating a computer program' (417, my italics). And from this, Searle concludes that programs produced by workers in artificial intelligence can never provide an adequate theory of any intentionality.

Searle's aim is to dispute certain strong claims from the field of AI: (1) that a particular computing machine literally understands English, (2) that it has this intentional attribute in virtue of the program it is running, and (3) that the structure of that program therefore reveals something about how the soul understands English.

Now it seems that many of Searle's critics have not understood his aim. He has been criticised as if he had been arguing that 'computers cannot think.' No such conclusion is his goal at all. In fact he states that '*only* a machine could think, and indeed only very special kinds of machines, namely brains and machines that had the same causal powers as brains' (424).

However, there are problems with Searle's arguments. For one thing, it is not completely clear what 'causal properties of the brain' he has in mind. For another, there is an ambiguity in his phrase 'instantiate a computer program.' The ambiguity is the same as that in 'follow a rule' and 'obey a law.' Strong sense: a man, such as Searle himself in his thought-experiment, might *literally follow* a rule or a set of instructions, in that he intentionally acts in accordance with such instructions, which he understands and is aware of. Weak sense: a group of billiard balls might *exhibit a regularity* expressed by e.g., the law of conservation of momentum, and in that sense they could be said to obey the law of conservation of momentum.



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

The question then is which sense of 'instantiating' is meant in Searle's claim that the brain does not produce intentionality by instantiating a program. More generally, in which sense do computing machines instantiate programs? I believe that one problem in discussing minds (souls, actually) and machines has been a failure to take this distinction seriously, and to assume that there is something 'in between' these two sorts of instantiating – a sort of 'unconscious but literal' following of rules.

Related to this problem is a lack of clarity on Searle's part about what does and does not follow from NAS (which he does undeniably establish). I will point out the importance of the words 'by itself' in NAS, as well as problems that would be involved in trying to show something about *necessary* conditions for having various intentionalities. I will also note that Searle's own version of NAS, which has 'never' where I have 'not always,' is too strong and does not follow from his example.

## I. The Example and Theorem

Searle's argument has three parts: (1) the Chinese Room Example, (2) a conclusion that seems to follow from it, which I shall call the Chinese Room Theorem, and (3) the supposed implications of the Theorem. Here is the Chinese Room Example: imagine that John R. Searle is locked in a room, and that he has a pack of cards with Chinese characters printed on them and an instruction book. The instructions in the book, an AI program, specify which cards are to be pushed out of the room through a slot, as a function of cards that are going to be pushed in. Then, some cards with Chinese characters are pushed in, and Searle follows the instructions in the book and pushes some cards out. From outside the room then, it seems that the man in the room understands Chinese. For the cards that were pushed in tell a story in Chinese about restaurants, and then ask questions in Chinese about the story, and the cards that come out give sensible and grammatical replies in Chinese. But the man in the room, Searle himself, in fact does not understand Chinese. Thus, *S* simulates being in state *C* by following an algorithm, but *S* is not in state *C*. That is the Chinese Room Example.



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

I would revise Searle's example slightly, by supposing that the instruction book states everything in terms of positive integers written on the backs of the cards, so that he does not need to pay any attention to the Chinese characters on the front. This makes it clearer that even if Searle 'internalizes' the rule book, he still has not learned Chinese. It is consistent with the ways that symbols are represented within a computer, and it serves to remind us of the Skolem-Löwenheim theorem and of the opacity of representation and intentionality.

Douglas Hofstadter complained that Searle ignores the fact that he would be very, very slow in producing his responses – much slower than a computer or a speaker of Chinese (433). But suppose that Searle is a calculating prodigy and computes his responses very quickly. (Remember, everything is represented as arithmetic.) Is this possible? Of course it is.

In fact, suppose that for any sequence of cards pushed into the room, the AI program in question produces the same response that a real Chinese speaker would produce. Then aren't real speakers of Chinese just such calculating prodigies? (Or, at any rate, doesn't AI dogma claim that they are?)

Calculating prodigies generally use a variety of tricks. In this case, a real Chinese speaker locked in the room might compute the complicated numerical function by interpreting the sequences of positive integers coming into the room as stories and questions in Chinese about restaurants. Searle himself might develop a different trick.

The Chinese Room Example challenges the claims by AI enthusiasts that a computer responding in English to English inputs actually understands English, and does so by executing a program – for we see that an entity might follow an algorithm and simulate having an intentional attribute *C*, without actually having that intentional attribute *C*. Hence, the Chinese Room Theorem:

CRT: There do not exist a program, a computer, and an intentional attribute such that the computer can come to have the attribute *just by* instantiating the program.

For just pick a person who lacks the attribute, and have him follow the



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

program, on the model of Searle in his room.[1] The person will still lack the intentionality; thus, the power to produce that intentionality is not in the program. At least the power to produce that intentionality *in humans* is not in that program. CRT follows if we can infer that since this is true of humans, it is also true of computers. Is there some doctrine that strong AI is committed to that would force its defenders to allow this inference?

Now Searle certainly does not claim that instantiating a program is sufficient for *lacking* any particular intentionality. If *I* were locked in the Chinese Room following that algorithm, I would not understand Chinese *by* following the algorithm, but I do understand Chinese, and I would continue to do so while following the algorithm. Following the algorithm does not cause me to stop knowing Chinese. This leaves it open that a computer might understand English or Chinese. The significance of the Theorem is supposed to be that even if a computer does understand Chinese, its program does not *explain* how it does this; however, I shall argue that the Chinese Room Theorem has no such implications.

## II. What Programs and Computers Do

The old distinction mentioned above (between intentionally following a rule and merely behaving in accordance with one) has application here, because computers do not literally execute programs! Searle, in the room, is executing instructions, but computers just do not do any such thing.

Many people believe that computers do execute instructions. They believe this because the popular press says so, because computer science textbooks say so, and because the experience of apparently

---

1 This will not work if the state in question is just plain thinking. If I merely claim that a certain program, running on a computer, produces thought, there is no Chinese Room Example to refute this, since the man in the room has to think to literally follow the program.



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

giving instructions to a computer, which it then apparently executes, is a quite common one.

A minor bit of sophistication appears in computer science textbooks. Typically, it is explained that a program that you write in a programming language, such as Fortran or Algol, has to be translated into another language, a machine language, that the computer can 'understand.' Then, so the explanation goes, the computer will execute those machine-language instructions (after also including needed subroutines, etc.). But the computer user generally does not need to worry about this, because the computer itself will do that job of translating.

So, when you enter a sequence of instructions such as

```
DO (N=1,100)
PRINT N, 'I WILL NOT PUSH IN LINE'
FIN
```

the computer does not execute *those* instructions. It translates them into a sequence of perhaps three times as many machine instructions, and then, supposedly, it executes those.

How? Obviously, nothing worth mentioning has been explained yet. The above 'explanation' is metaphorical and question-begging. It leaves us still in the dark about the details of the process by which the computer supposedly reads an instruction and then carries it out. That this explanation is worthless and incoherent seems to have gone unnoticed.

The solution to this problem is that computers just do not do any such thing. They do not execute instructions, and they do not execute sequences of instructions (i.e., programs). This is so whether the instructions are in a quasi-natural algorithmic programming language or in a so-called machine language. (For an example of this confusion, see Smythe's reply, p. 448, col. 2.) Now computers do *appear* to execute instructions, and we can *represent* their behaviour as that of executing a set of instructions. But literally they do no such thing.

Let me elaborate. Many years ago my father and I built a little machine for solving syllogisms. The front of the machine had two



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

wheels and a row of lights. Wheel 1 could be turned to any one of several positions, representing forms in '*A*' and '*B*,' such as 'all *A* is *B*,' 'some *B* is not *A*'; wheel 2 was similar, except that its positions represented forms in '*B*' and '*C*.' The lights were labeled with forms in '*A*' and '*C*,' and one was labeled 'no conclusion.' Setting the wheels to two premises resulted in the appropriate light going on.

How? Was the machine following instructions? No. The back was a maze of wires, and the principles involved were the elementary ones of series and parallel circuits.

One could build a machine for playing tic-tac-toe (noughts and crosses) or chess in just the same way. First, draw a wiring diagram, and then get out your soldering gun and some wire. It is in fact quite reasonable to imagine building a tic-tac-toe machine this way. However, the wiring diagram for a decent chess-playing machine would be so complicated that no mere human would be able to draw it in the first place.

This is where programs enter. A task such as deciding a chess move or calculating a cosine is described as a sequence of discrete steps. Suppose then that we had a device that could automatically turn this description into wiring. Then we would have a much easier job of building a chess machine: we would write a program and feed it to this device, and it would build the machine. Of course the program has to be written in a formalized language and represented in a way that will cause physical action to take place. That is easy to arrange — represent it by holes punched on paper tape, like a player piano roll.

Now we have (a) our algorithm, (b) its representation as a sequence of holes punched on tape, (c) the machine that the tape is fed to, and (d) the machine that is then constructed as a result. The machine and wiring that result might be made small — burned into a silicon chip, perhaps. The resulting machine is identical in kind with one built directly from a wiring diagram. Neither executes programs. Neither contains instructions. In fact this is exactly how chess-playing microcomputers are made.

Programs then have two quite distinct uses: (1) to build complex machines, and (2) to represent complex behaviour. The machines that are built with them do not themselves execute instructions. In use (2), as a representation of what a machine or a person is doing, a program is often easy for humans to use and understand. But for this represen-



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

tational use, the program does not need to be written in a formal programming language at all – a flow chart or informally stated algorithm will do. Now although such an algorithm can be a useful representation of what a machine is doing, computers do not literally execute programs. Searle, locked in the room, is literally following an algorithm, but this is a very special case. Most of the time, neither people nor machines are literally following programs.

I have not said that it is *impossible* for a computer to literally follow instructions. If a computer were built with the 'causal powers of the brain,' then possibly it could understand English and thus literally follow instructions written in English. And then, we could give it Searle's book of instructions and lock it in a room with that book and the cards with the Chinese characters... . It would understand English, but it would still not understand Chinese!

As another example of programs as representations, consider a system of self-service elevators in a large hotel. By sufficient playing with the system, we could produce a written program that would exactly represent the behaviour of the system. ('Send nearest elevator to answer call unless going in opposite direction, in which case... unless... etc.') But even if this sequence of instructions (i.e., this program) is a perfect representation of the elevator system's behaviour, it does not follow that the system is executing any instructions at all. (Newer elevator systems have parts of their wiring actually generated via programs, i.e., they are controlled by microcomputers, but this is clearly irrelevant.)

In which sense do actual computing machines 'instantiate' or follow or execute programs? Only in the weak sense: they exhibit regularities which can be expressed by algorithms written in formalized programming languages. In which sense does Searle himself in his thought-experiment instantiate a program? In the strong sense: he literally follows a sequence of instructions. The strong sense implies the weak, so Searle in the room also instantiates the algorithm in the weak sense – his behaviour accords with it.

Art is representation aimed at beauty. Science is representation aimed at explanation. To describe a thermostat or a player piano or a computer as following instructions might be pretty and useful, but it is not a good explanation of anything, and therefore it is not good science. It is a gross confusion to say that a hole in a certain place in a



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

piece of paper is a player-piano-language instructing meaning 'hit middle C.' There just is no player piano language, and there just is no machine language for any computer. A player piano does not follow instructions. Neither does an elevator system or a computer. A program is a *representation of* what such machines are doing.

## III. Some Remarks about Representation

It is easy to confuse representing with being. Is this dot on the map Wellington, New Zealand? Of course not – the dot is much too small. Is the number two the set {0, {0} }? Of course not – the set has two members, and the number two does not. Of course that set might be used to *represent* the number two. Representation is one of the most important notions in the history of philosophy, mathematics, and in the sciences and arts generally.[2]

J.L. Synge has a nice name for the error of mistaking a representation of something for the real thing. Synge calls this the *Pygmalion Syndrome*, after the mythological king who fell in love with a statue. Describing a computer as executing instructions is an example. So is thinking that the number two could really be {0, {0} }.

Pygmalion Syndrome is obviously epidemic among workers in the field of so-called artificial intelligence. As Searle points out, nobody would confuse a computer simulation of a storm with a real storm, but computer simulations of intelligence are often described as intelligent. So-called artificial intelligence is probably only *simulated* or *imitation* intelligence rather than genuine artificial intelligence.[3]

---

2 Exercise: read *Republic* VI 509d-511e; then rewrite the last chapter of *Word and Object*, putting 'represent by' for 'identify with' throughout. Possibly the only real disagreement between Quine and Plato is simply over whether our understanding of abstract ideals is clearer or muddier than our understanding of symbols for them.

3 Being artificial implies being actual: AI can get you just as pregnant as the natural sort (AI = artificial insemination). A simulated insemination of you might be run on a computer, but you would not need to worry about it any



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

The mindless reply to Searle by Douglas Hofstadter is a good example of Pygmalion Syndrome gone wild. Hofstadter takes the entire question to be one involving merely a conflict of religious dogma. He then tells us that he specifically counts himself among those who 'find in reductionism the ultimate religion' (434). Reductionism, at its worst, is the belief that if you have constructed a representation of something (e.g., points and lines) in terms of something else (e.g., classes, real numbers, and ordered pairs), then the something actually is the something else. This is obviously just more Pygmalion Syndrome.

And intelligence is one thing, consciousness another. Attributing intelligence to something, person or machine, seems independent of attributing consciousness or intentionality to it. I am even willing to be totally behaviouristic about intelligence – but not about intentional states, such as understanding Chinese.

In any event, Searle is not presenting any argument about the limits of *simulated* intelligence. There may be programs that pass the Turing test. If not, wait until next year.

Searle's Chinese Room Example does establish without doubt that the Turing test is invalid. But the Turing test should not be thought of as a test of *intelligence* in the first place. If we want a test of intelligence, we need not look far. We already have standard intelligence tests, taken by thousands of high school students every year. I suggest then this challenge to AI: write a program that will consistently get decent scores on these standardized tests. (No complaints about cultural bias, please!)

---

more than you need to worry about someone's throwing darts at your photograph.

Being actual is not quite the same as being real; if someone dyes his grey hair black, the actual colour is then black, but the real colour is grey; 'real' can mean 'original'. 'Artificial' can sometimes seem to mean 'imitation', as in 'artificial flowers', but I find this a bit of a misnomer. An imitation hand merely has to look like a natural hand, but an artificial hand must do the work of a natural hand, even if it does not look like one. If the only work required of flowers is to look flowery, then that would explain why imitation ones are called artificial. A plastic imitation tree could be an artificial Christmas tree, but it could not be an artificial apple tree, although it might be an imitation apple tree.



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

## IV. Programs as Theories

But what of the Chinese Room Theorem? The Chinese Room Example shows that instantiating (in the strong sense of literally following) a set of instructions is not *always by itself* sufficient for any intentionality. Does it follow that for anything that does have a particular intentionality *C*, such as an understanding of Chinese, it does not have it *by* following any program or algorithm? Certainly not. Turning the key in my ignition switch is not *by itself* sufficient for starting my car, as I am occasionally reminded by a dead battery. Does it follow that I never start my car *by* turning the key? Of course not. When other necessary conditions are satisfied, I do start my car by turning the key.

It is for this reason that I have put the word 'just' in my version of the Chinese Room Theorem. Turning a key is not *by itself* sufficient for starting my car; therefore, the explanation of my car's starting is not *just* that I turn a key. But so what? My turning the key is still an important *part* of the explanation of my car's starting. So just what is the significance of the Chinese Room Theorem? Notice that it leaves wide open the possibility that literally following a computer program could still be an important *part* of the explanation of intentionalities. What Searle has shown is that instantiating a program cannot always be the *whole* story. This may not seem to be much — but remember, his target is only the excesses of what he calls 'strong AI.'

Some of the locutions used by Searle are worth a comment. The following are equivalent: *A* is not by itself a sufficient condition for *B*; *A* is not always a sufficient condition for *B*; *A* is not a sufficient condition for *B* unless *C* (for some particular *C*); *A* is a sufficient condition for *B* only if *C*; *A* is at most a *proper part* of the explanation of *B*; *A* is not the whole of the explanation of *B*. And then there is a second group of locutions, distinct from these, but equivalent to each other: *A* is not by itself a necessary condition for *B*: *A* is not always a necessary condition for *B*; *A* is not a necessary condition for *B* unless *D* (for some particular *D*); *A* is a necessary condition for *B* only if *D*; *A* is not necessary to explain *B*; *A* is not the only possible explanation of *B*.

It is true that for every program *P* and every intentional state *C*, literally following (strong sense) program *P* is not enough to guarantee



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

being in state *C*, since a man in a room could follow the program without being in the state. From this, and the fact that literally following (strong sense) a program *P is* sufficient for instantiating *P* in the weak sense, it does follow that instantiating (weak sense) a program *P* is not enough to guarantee being in *C* either. And *of course* nothing can have *C* (e.g., an understanding of Chinese) *by, through,* or *in virtue of* exhibiting the regularity represented by a program.

Consider this parallel to Searle's argument. A quadrilateral (a man in a room) can be equi-angular (can instantiate a program *P*) without being equilateral (without understanding Chinese); therefore, being equi-angular is 'not always by itself sufficient' for being equilateral. But it just does not follow that being equi-angular is *never* sufficient for being equilateral. For *triangles,* being equi-angular is sufficient for being equilateral. What is true of quadrilaterals (i.e., that being equi-angular is not by itself sufficient for being equilateral) is not true of triangles.

Now let us put 'computer' or 'brain' in place of 'triangle' in this argument. We see that the Chinese Room Example shows NAS, but does not generalize to the Chinese Room Theorem. NAS is fairly weak; it says that instantiating a program is *not always* by itself sufficient. Searle's own version said that instantiating a program is *never* by itself sufficient for intentionality. But that does not follow from the Chinese Room Example. (For further discussion of this point see Sharvy 1983.)

Consider the complex property of *being such that instantiating an AI program that simulates an understanding of Chinese is not by itself sufficient for actually understanding Chinese.* Humans have this property, as the Chinese Room Example demonstrates. Then if all computers were humans, all computers would have it also, and we could derive the general Chinese Room Theorem. But even strong AI does not hold that all computers are humans. Searle needs to give us a precisely formulated dogma of strong AI that he can use against AI by showing that the dogma allows the inference from his example to the general theorem.

It is interesting that Searle does not attempt to show *converse* examples. Why hasn't he provided us with a man who understands English but does not literally follow any AI algorithms, to convince us that following algorithms is *not always even part* of the explanation of



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

understanding English. To show that something is not always part of an explanation of something, one needs to show that it is not a *necessary* condition! In the present case then, it is the fact that some of us speak Chinese, but do not at the same time literally and consciously push addresses on stacks that shows that we do not speak Chinese *by* literally and consciously following AI algorithms.

For another example, suppose that I wanted to prove that I do not keep my balance on my bicycle *by* calculating solutions to partial differential equations. I would show this by riding my bicycle, while at the same time paying careful attention to whether or not I was also calculating solutions to partial differential equations. (To build a computer that would actually balance a bicycle, one would need to write a program that did include routines to do such calculations. But recall sec. 2 above; computers do not compute.) The fact that I balance the bicycle without doing the calculations shows that doing such calculations is not always a *necessary* condition for balancing the bicycle; and this shows that I do not always balance bicycles even in part *by* doing any such calculations.

This argument is similar to one given by Berkeley (sec. 12) against the view that we judge distance in part by using optical angles and trigonometry. But the question raised here is precisely one involving introspection and the possibility of following rules literally yet unconsciously. Berkeley just says that he above anyone else is the best judge of whether or not he is doing trigonometry problems, and he says that he isn't doing any such thing.

But is this sufficient to prove the point? Why can't we say that Berkeley was (literally but unconsciously) doing such computations?[4] Is there some way of literally, but not consciously, following these algorithms? Is there something in between literally and consciously obeying a rule and merely behaving in accordance with it? Does it

---

4 Those who think this makes no sense should consider the following game: players alternate selecting a card from an exposed pack containing nine cards, numbered 1-9. The first player to be able to make a total of exactly fifteen, using exactly three of the cards he has taken, is the winner. I claim that everyone has played this game hundreds of times, but without being aware of it (see Sharvy 1985).



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

make sense to say that some *part* of me literally *and consciously* follows algorithms, but that *I* am unconscious of this part?

Now Searle, when conversing in English, does not do so by consciously and literally following or obeying any instructions. But he is exhibiting regularities which can be represented by a large number of distinct algorithms. Thus, he is no different with respect to English than a computer that appears to converse in English. Both instantiate (weak sense) many programs; neither literally follows (strong sense) any program. The possibility of artificial intentionality remains an open question; but Searle was never attacking this in the first place. The point then is that whether or not metallic machines are ever built which possess intentionality, they will not do so *by* literally following programs either. But why should they? 4.2 billion *organic* 'machines' don't.

Instantiating a program *P* in the weak sense merely involves exhibiting a certain regularity expressed in *P*. Program *P may* fail to express a theory that explains what it is to be in state *C*; that is, *P may* fail to be a representation that explains *C*. But this fact is merely an instance of the more general truth of philosophy of science, that not every statement of a regularity constitutes an explanation.

Searle wanted to show that programs which cause computers to simulate a knowledge of English do not *explain* what it is to know English. He wanted to dispute the claim that AI programs 'explain human cognition' (417, col. 1) and 'human understanding' (418, col. 1). But this is a large step on Searle's part, which I believe is not justified. There might well be AI programs that explain and accurately represent what is involved in understanding English, even though a man could be locked in a room following the program without understanding English. Some regularity statements explain, and some do not. What Searle needed was a specific argument that AI programs merely describe regularities, but do not *explain* e.g., what it is to understand Chinese.

Of *course* the explanation of how I understand English is not that I do it *by* literally and consciously following an algorithm. But it is consistent with this that an explanation of how I understand English might be expressed and represented by an algorithm. Perhaps one of the many algorithms that expresses the regularities I exhibit in conversing in English also constitutes a good *theory* of what it is to under-



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

stand English. The actual elements of the program could explain and reveal the structure of the soul's understanding of English. For example, the program might represent tree structures, recursive syntactic rules, etc. And the situation here is not peculiar to mental activity or intentionality. Billiard balls do not behave as they do *by* literally following the laws of physics. And they certainly do not behave as they do *by* exhibiting the regularities described in those laws, i.e., by behaving as they do. In fact, billiard balls do not behave as they do *by* doing anything. But those laws may still provide explanations of the motions of billiard balls.

The situation we are left with is familiar in the history of science. Something rule-like fits the observed data and accurately predicts future data. It is a useful representation of part of the world. But is it an *explanation*? Against strong AI claims, we simply note that successful simulations do not warrant claims that the representations therein provide such explanations. A theory can work, but still be a bad one. A theory can make accurate predictions, yet explain nothing.

And against Searle: suppose that we had a machine that was constructed on principles isomorphic with that of the human brain, but which could have patterns imprinted upon it via programs via compilers.[5] Would not Searle have to admit that such an item might literally understand English, despite the fact that a program was used in its construction? Yes (see Lycan's commentary, 435). And now the functionalist worry: consider a machine that was *not* constructed exactly like a human brain – consider a Martian's brain – don't we want to admit that these might have the same intentionalities that we

---

5 The relevant 'causal powers of the human brain' that Searle mentions should not involve its being made of carbon, nitrogen, etc. They should be the powers that come from the structure of the individual neurons, and their connections (422, col. 2). Of course a computer could simulate a neuron, or even a network of ten billion of them, but that would still be just a simulation of a brain. It might not be a real brain, and it might not have all the relevant causal powers of a real brain.

Unfortunately, there is now empirical evidence that the brain may have very little to do with thinking after all. John Lorber, a neurologist at Sheffield University, has found a number of normal, intelligent people who have virtually no brains at all (see Paterson).



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press

do? Yes? How about a PDP-10? And if a PDP-10 came to literally understand English, wouldn't we look to the program for some understanding of how it did that? Yes. Of course that program might not provide an explanation of *human* understanding of English. And then again, it might.

Received December, 1982

**BIBLIOGRAPHY**

Berkeley, George (1709), *Essay Towards a New Theory of Vision*

Hofstadter, Douglas R., 'Reductionism and Religion' (comment on Searle). *BBS* 3 (1980) 433-4

Lycan, William G., 'The Functionalist Reply (Ohio State)' (comment on Searle). *BBS* 3 (1980) 434-5.

Paterson, David., 'Is Your Brain Really Necessary?' *World Medicine* 15, (1980) 21-4

Plato, *Republic* VI

Quine, W.V. *World and Object*, Chap. VII. (Cambridge MS.: MIT 1960)

Searle, John R. 'Minds, brains, and programs.' *The Behavioral and Brain Sciences* 3 (1980) 417-57 (includes commentaries and replies). Reprinted in Douglas R. Hofstadter and Dandiel C. Dennett, eds., *The Mind's I*, (New York: Basic Books, 1981) 353-73; and in John Haugeland, ed., *Mind Design*, (Cambridge, MS.: MIT Press, Bradford Books, 1981) 282-306.

Sharvy, Richard. 'It Ain't the Meat, It's the Motion.' *Inquiry* 26 (1983) 125-31.

Sharvy, Richard, 'A Philosophy Experiment.' *Teaching Philosophy* (1984) forthcoming

Smythe, William E., 'Simulation Games' (comment on Searle). *BBS* 3 (1980) 448-9

Synge, J.L., *Talking about Relativity*. (Amsterdam: North Holland 1970)



Copyright (c) 2005 ProQuest Information and Learning Company
Copyright (c) University of Calgary Press